Einführung in (Binäre) Bäume

Marc Rennhard
http://www.tik.ee.ethz.ch/~rennhard
rennhard@tik.ee.ethz.ch

Bedeutung und Ziele

Bäume gehören ganz allgemein zu den wichtigsten in der Informatik auftretenden Datenstrukturen, die Sie selbst schon oft angetroffen haben. Insbesondere Binärbäume haben die Eigenschaft, dass Daten effizient eingefügt, gefunden und gelöscht werden können.

Nach dieser Lektion können Sie Folgendes:

- Sie wissen, was Bäume in der Informatik sind
- Sie kennen die Terminologie von Bäumen und können sie richtig anwenden
- Sie kennen einige Anwendungsgebiete von Bäumen in der Informatik
- Sie wissen, was Binärbäume sind und kennen deren wichtigste Eigenschaften
- Sie kennen die drei Traversierungsmethoden von Binärbäumen und können sie richtig anwenden

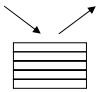
Ablauf

- Kurzer Rückblick
- Allgemeine Betrachtung von Bäumen
  - Terminologie
  - Lernkontrolle zur Terminologie
  - Definition eines Baumes
  - Beispiele von Bäumen in der Informatik
- Binärbäume als wichtiger Spezialfall allgemeiner Bäume
  - Definition und Eigenschaften von Binärbäumen
  - Traversieren von Binärbäumen
  - Lernkontrolle zum Traversieren

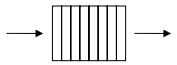
Rückblick

Bis jetzt haben Sie folgende Datenstrukturen kennengelernt:

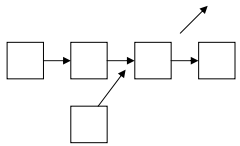
- **Stack**: LIFO (last-in-first-out) Speicher



- **Queue**: FIFO (first-in-first-out) Speicher



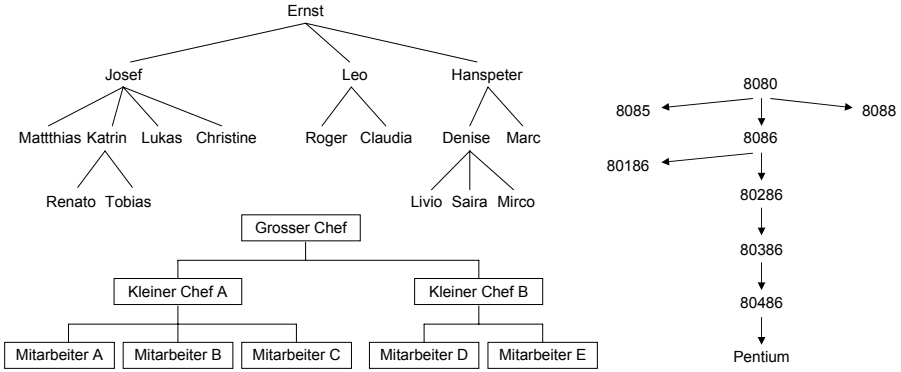
- **Liste**: Beliebiger Zugriff auf einzelne Elemente



Dies waren alles lineare Datenstrukturen, denn jedes Element hat immer maximal einen Vorgänger oder Nachfolger

- Mit Bäumen lernen Sie heute eine nicht-lineare Datenstruktur kennen

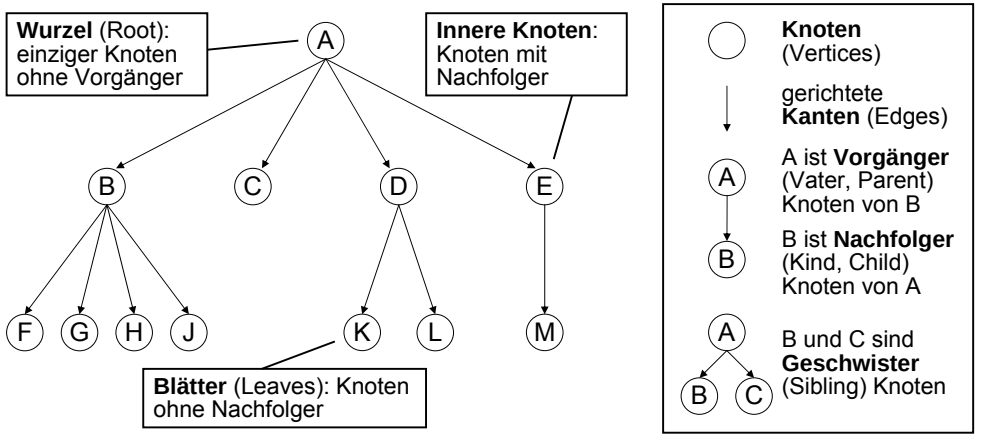
Bäume findet man fast überall



Typische gemeinsame Eigenschaften all dieser Bäume:

- Mehrschichtige Hierarchie von Elementen mit einem eindeutigen Ursprung
- Jedes Element (ausser dem Ursprung) ist zu genau einem Element in der darüber liegenden Herarchieschicht zugeordnet und kann einen Bezug zu mehreren Elementen in der darunterliegenden Hierararchieschicht haben

Terminologie (1) und Definition



Definition eines Baumes (informell):

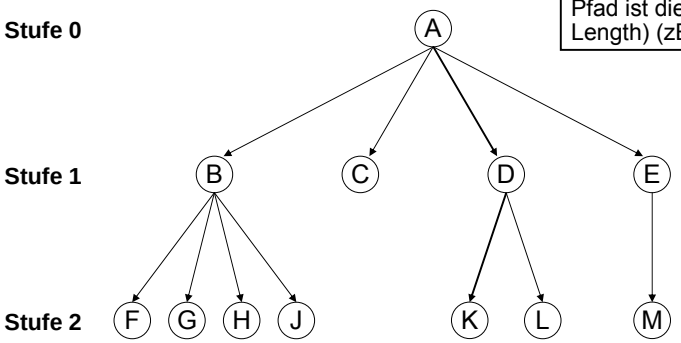
Ein Baum besteht aus Knoten und gerichteten Kanten, wobei eine Kante jeweils zwei verschiedene Knoten verbindet. Die Wurzel ist der einzige Knoten ohne Vorgänger; alle anderen Knoten haben genau einen Vorgänger. Alle Knoten haben eine beliebige Anzahl von Nachfolgern.

Terminologie (2)

Die Knoten mit der gleichen Pfadlänge k befinden sich auf der gleichen **Stufe k**

Es gibt genau einen **Pfad** (Path) von der Wurzel zu jedem anderen Knoten (zB A-D-K)

Die Anzahl der Kanten in einem Pfad ist die **Pfadlänge** (Path Length) (zB 2 für A-D-K)



Die **Tiefe t** eines Baumes ist definiert als die maximale Pfadlänge + 1 (Hier 2 + 1 = 3)

Lernkontrolle Terminologie

Beantworten Sie die folgenden Fragen zum Baum rechts:

Welche(r) Knoten ist/sind die

- Wurzel:_____
- Inneren Knoten:_____
- Blätter:_____

Vorgänger von L:_____

Nachfolger von D:_____

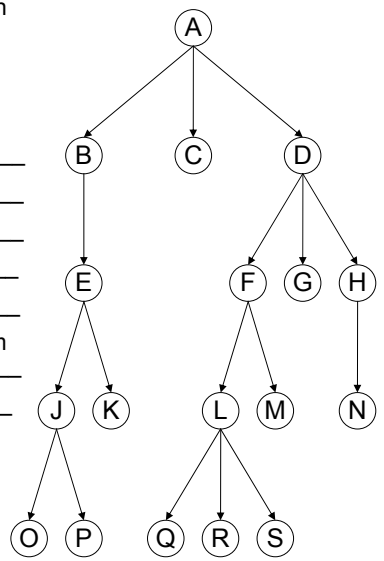
Geschwister von B:_____

Welche Knoten liegen auf dem Pfad von A nach Q:_____

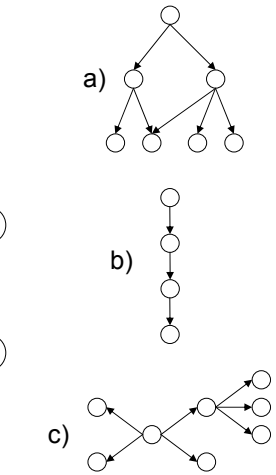
Wie lange ist dieser Pfad?____

Auf welcher Stufe befindet sich L?_____

Tiefe des Baums:_____



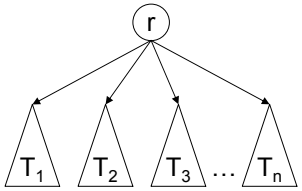
Welches sind gültige Bäume?



Formale Definition

Rekursive Definition:

Ein Baum T ist entweder leer oder besteht aus einer Wurzel r mit 0…n Teilbäumen $T_1 \ldots T_n$



Aus dieser Definition folgt:

- Jeder beliebige Knoten in einem Baum kann als Wurzel eines neuen Baumes betrachtet werden.
- Aus 1…n beliebigen Teilbäumen $T_1 \ldots T_n$ erhält man einen neuen Baum, indem man die Wurzeln dieser Teilbäume zu den Nachfolgern einer neuen Wurzel macht.

(Nicht-Rekursive Definition (vergleich mit Definition von Graphen):

Ein Baum T=(V,E) besteht aus einer Menge von Knoten V und einer Menge von gerichteten Kanten E. Die Wurzel $r \in V$ hat nur Ausgangskanten; alle anderen Knoten haben genau eine Eingangskante. Für alle Kanten $e \in E$ gilt $e = (v_1,v_2)$, wobei $v_1,v_2 \in V$ und $v_1 \neq v_2$.)

Baum als verallgemeinerte Liste

Knoten einer einfach verketteten Liste:

```
public class ListNode {
  object data;
  ListNode next;

  ListNode(Object d) {
    data = d;
  }
}
```

Knoten eines Baumes:

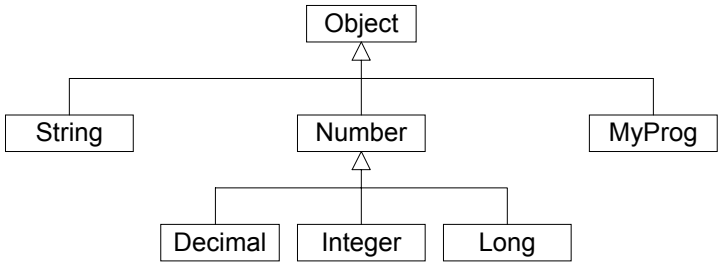
```
public class TreeNode {
  object data;
  LinkedList next;

  TreeNode(Object d) {
    data = d;
  }
}
```


Beispiele Bäume in der Informatik (1)

Die Klassenhierarchie in Java kann in einem Baum dargestellt werden, denn:

- Die Klasse „Object“ ist die Wurzel der Klassenhierarchie
- Alle anderen Klassen haben genau eine Superklasse



Aber:

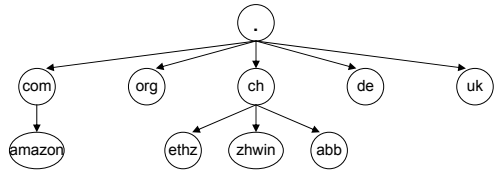
- Gilt nicht für jede OO-Sprache (zB C++ mit Mehrfachvererbung)

Beispiele Bäume in der Informatik (2)

Dateien in einem Filesystem werden oft in einem Baum angeordnet

- Das Wurzelverzeichnis (Root Directory) entspricht der Wurzel
- Verzeichnisse entsprechen den inneren Knoten
- Dateien entsprechen den Blättern
- Das Navigieren durch das Filesystem entspricht dem ab- und aufsteigen entlang einer Kante
- Aber: mit Mechanismen wie Links in Unix oder Shortcuts in Windows entspricht das Filesystem eigentlich nicht mehr einem Baum!

Domain Names im Internet (zhwin.ch, abb.ch, amazon.com) sind in einem Baum angeordnet



Fragen zu Bäumen im Allgemeinen?

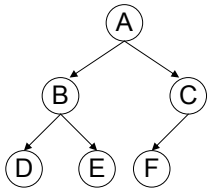
TIK
ETH Zurich

Binäre Bäume (Binärbäume)

TIK
ETH Zurich

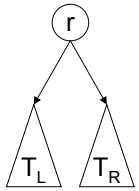
Definition als Spezialfall eines allgemeinen Baumes :

Ein Binärbaum ist ein Baum bei welchen ein Knoten maximal zwei Nachfolger hat. Dabei ist jeder Knoten ausser der Wurzel eindeutig als linker oder rechter Nachfolger seines Vorgängers gekennzeichnet.



Rekursive Definition Binärbaum:

Ein Binärbaum T ist entweder leer oder besteht aus einer Wurzel r mit einem linken und einem rechten Teilbaum T_L und T_R



Vom Baum zum Binärbaum

TIK
ETH Zurich

Knoten eines Baumes:

```
public class TreeNode {
  object data;
  LinkedList next;

  TreeNode(Object d) {
    data = d;
  }
}
```

Knoten eines Binärbaumes:

```
public class BinaryTreeNode {
  object data;
  BynaryTreeNode left;
  BynaryTreeNode right;

  BinaryTreeNode(Object d) {
    data = d;
  }
}
```


Wozu eigentlich Binärbäume?

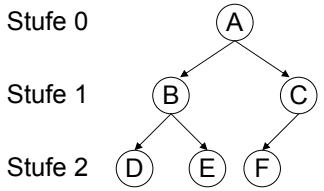
TIK
ETH Zurich

- Bisher haben wir allgemeine Bäume betrachtet, die sich vor allem zur allgemeinen Darstellung und hierarchischen Strukturierung von Daten eignen
- Binärbäume sind durch die Einschränkung auf maximal zwei Nachfolger eines Knotens dafür nicht geeignet
- Die Datenstruktur „Binärbaum“ wird vielmehr dazu verwendet, Daten so zu organisieren, dass sie effizient eingefügt, gefunden und gelöscht werden können

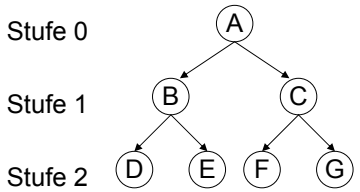
➔ Um dies zu erkennen, müssen wir einige spezielle Eigenschaften von Binärbäumen anschauen

Eigenschaften

- Auf Stufe k befinden sich maximal 2^k Knoten
- In Binärbaum heisst voll, wenn alle Stufen k ausser der untersten komplett besetzt sind, d.h. jeweils 2^k Knoten besitzen



- In eine Stufe k passt immer genau ein Knoten mehr als in allen darüber liegenden Stufen 0…(k – 1) zusammen ➔ ein Binärbaum der Tiefe t besitzt maximal $2^{t-1} + (2^{t-1} - 1) = 2^t - 1$ Knoten
- Daraus folgt, dass ein voller Binärbaum mit n Knoten eine Tiefe von $\lceil \log_2 (n + 1) \rceil$ hat



Die Tiefe eines vollen Binärbaumes wächst also logarithmisch mit der Anzahl der Knoten ➔ dies bildet die Basis damit Daten im Binärbaum effizient eingefügt, gefunden und gelöscht werden können (mit Aufwand O(log n), aber mehr dazu folgt später)

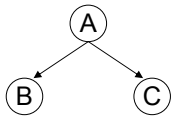
Traversieren eines Binärbaumes

Problem: Man möchte alle Daten, die in einem Binärbaum gespeichert sind, durchgehen; zum Beispiel um sie als Liste auszugeben oder um ihre Anzahl oder ihre Summe zu berechnen

➔ Mit Traversieren eines Binärbaumes wird das systematische Besuchen all seiner Knoten bezeichnet

Es gibt viele verschiedene Reihenfolgen, in welchen man die Knoten eines Binärbaumes ausgeben kann, zum Beispiel:

- A, B, C
- A, C, B
- B, A, C
- B, C, A
- C, A, B
- C, B, A



Die drei wichtigsten Reihenfolgen werden mit Preorder, Inorder und Postorder gekennzeichnet und lassen sich rekursiv definieren

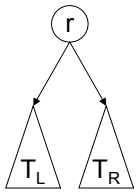
Traversierungsalgorithmen

Traversieren eines Binärbaumes in Preorder:

1. Besuche die Wurzel
2. Durchlaufe den linken Teilbaum in Preorder
3. Durchlaufe den rechten Teilbaum in Preorder

Traversieren eines Binärbaumes in Inorder:

1. Durchlaufe den linken Teilbaum in Inorder
2. Besuche die Wurzel
3. Durchlaufe den rechten Teilbaum in Inorder

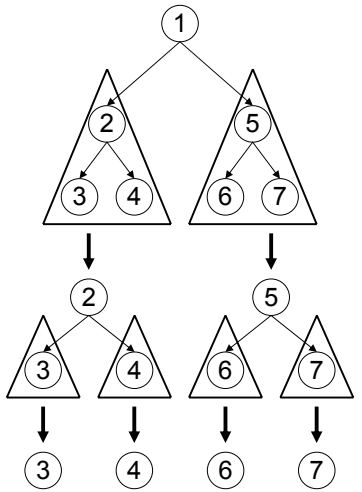


Traversieren eines Binärbaumes in Postorder:

1. Durchlaufe den linken Teilbaum in Postorder
2. Durchlaufe den rechten Teilbaum in Postorder
3. Besuche die Wurzel

Beispiel: Traversierung in Preorder

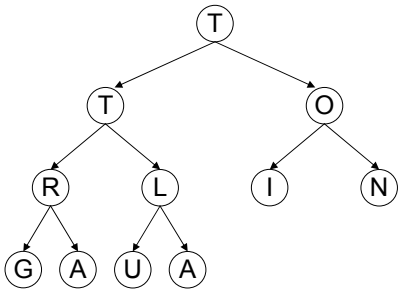
1. Besuche die Wurzel ➔ 1
2. Durchlaufe den linken Teilbaum in Preorder
   1. Besuche die Wurzel ➔ 1, 2
   2. Durchlaufe den linken Teilbaum in Preorder
      1. Besuche die Wurzel ➔ 1, 2, 3
   3. Durchlaufe den rechten Teilbaum in Preorder
      1. Besuche die Wurzel ➔ 1, 2, 3, 4
3. Durchlaufe den rechten Teilbaum in Preorder
   1. Besuche die Wurzel ➔ 1, 2, 3, 4, 5
   2. Durchlaufe den linken Teilbaum in Preorder
      1. Besuche die Wurzel ➔ 1, 2, 3, 4, 5, 6
   3. Durchlaufe den rechten Teilbaum in Preorder
      1. Besuche die Wurzel ➔ 1, 2, 3, 4, 5, 6, 7



Lernkontrolle: Traversierung in Inorder

TIK
ETH Zurich

Traversieren Sie den unten stehenden Binärbaum in Inorder. Wie lautet die Reihenfolge der Knoten?



Fragen zu Binärbäumen?

TIK
ETH Zurich

Zusammenfassung

TIK
ETH Zurich

Bäume bestehen aus Knoten und gerichteten Kanten

- Alle Knoten ausser der Wurzel haben genau einen Vorgänger
- Ein Knoten kann mehrere Nachfolger haben

Allgemeine Bäume findet man in der Informatik häufig

- Zum Beispiel Klassenhierarchie in Java, Filesysteme und Domain Names im Internet

Binärbäume sind ein Spezialfall von allgemeinen Bäumen

- Ein Knoten hat maximal zwei Nachfolger, welche als linker und rechter Nachfolger bezeichnet werden

Mit Traversieren bezeichnet man das systematische Besuchen und Auflisten aller Knoten in einem Binärbaum

- Preorder, Inorder und Postorder bezeichnen die drei wichtigsten Traversierungsmethoden